Daniel Ortega and Robert Miranda

Prof. Dorsey

CPSC 479

22 December 2021

Final Project: An Urban Complex

Our project rendered a futuristic urban complex with surrounding trees and clouds. The four techniques we used were sweeps to generate the building, Perlin noise to create clouds, procedural animation to add motion to the clouds, and L-systems to create trees.

The sweep technique was used in our midterm project to generate seashell models. The final building was inspired by the paper *Seashell Architectures*. To support this extension, the sweep technique was modified to support custom scaling and radii functions; in seashells, the growth rate is logarithmic, so the functions were parameterized by a single variable. The script was also modified to support sweeps of non-closed curves.

Perlin Noise was used in the Open Shading Language shader which outputs turbulence. More specifically, we implemented a version of the turbulence function as described in *An Image Synthesizer*. This turbulence function was then used to procedurally generate clouds within a bounded volume. A script created Blender nodes which calculate a texture coordinate within the volume and then assign that coordinate a density value from 0 to 1, as determined by the turbulence function. This results in a relatively realistic cloud texture. Much of the difficulty for this portion was figuring out a way to optimize the rendering of the clouds. Granted, rendering is still slow due to the use of OSL and the Cycles rendering engine, but still faster than methods that we tried before arriving at the final iteration.

To procedurally animate the clouds, we added keyframes which control the 4th dimension of the turbulence function and the translation of the calculated texture coordinate. This required adding translation to the OSL shader. The creation of clouds was handled in a Python script which creates an option within the "Add Mesh" menu to add clouds, as seen in the image at the bottom of the screen. One issue was getting the animation of the clouds to look realistic. Eventually, we figured out that linear interpolation, rather than a Bezier curve, was better suited to controlling the procedural animation of the clouds. As a result, only two keyframes were needed: the first and last. This greatly simplified the Python script which adds the mesh.

Finally, L-systems were implemented to produce trees. The script supports push-down stochastic L-systems to generate similar, yet random, trees to populate the screen. The script also implements a `turtle` that walks the L-system. The tree is created via adding a mesh for each segment in the generated L-system word. Several existing L-systems presented in *The Algorithmic Beauty of Plants* were implemented to test functionality, but the L-system for the trees is original. It was especially designed so the trees would tend to lean towards the normal direction to simulate the effect of wind, tying the L-system generating to the procedural animation of the clouds.

One challenge in the project was the scope of the design. The modifications to allow the sweep function to be used more generally, and the L-system function allowed full control, but this made it difficult to achieve pleasing artistic results. Another challenge specific to the L-system was that because each segment is added as an individual mesh, deep recursion could crash the Blender program. Our solution was to design the L-system to minimize the number of recursions in the algorithm by adding different generators for different length branches.

